

## C++编程等级认证六级 2023 年 12 月理论 180 分钟

【单选题】( 每题 2 分 )

1、关于 C++类和对象的说法，错误的是？( )

- A、在 C++中，一切皆对象，即便是字面量如整数 5 等也是对象      B、在 C++中，可以自定义新的类，并实例化为新的对象  
C、在 C++中，内置函数和自定义函数，都是类或者对象      D、在 C++中，可以在自定义函数中嵌套定义新的函数

正确答案：D

2、有关下面 C++代码的说法，错误的是？( )

```
class Rectangle {  
private:  
    class Point {  
    public:  
        double x;  
        double y;  
    };  
    Point a, b, c, d;  
    double length;  
    double width;  
public:  
};
```

- A、C++中类内部可以嵌套定义类      B、在类中定义类被称为内部类，定义类的类被称为外部类  
C、内部类可以随便访问，不需要通过外部类来访问  
D、代码中 Point 被称为内部类，可以通过外部类 Rectangle 来访问，Rectangle::Point

正确答案：C

3、有关下面 C++代码的说法，正确的是？( )

```
2 | using namespace std;  
3 | class newClass  
4 | {  
5 |     public:  
6 |         static int objCounter;  
7 |  
8 | };  
9 | int newClass::objCounter = 2;  
10| int main()  
11| {  
12|     newClass classA;  
13|     newClass classB;  
14|     cout << newClass::objCounter << endl;  
15|     cout << classB.objCounter << endl;  
16| }
```

- A、第 14 行代码错误，第 15 行正确      B、第 15 行代码错误，第 14 行代码正确      C、第 14、15 两行代码都正确  
D、第 6 行代码可修改为 objCounter += 1

正确答案：C

4、有关下面 C++代码的说法，错误的是？( )

```
struct BiNode {  
    char data;  
    BiNode *lchid, *rchid;  
};
```

```

class BiTree {
private:
    BiNode *Creat();
    void Release(BiNode *bt);
    BiNode* root;
public:
    BiTree() {
        root = Creat();
    }
    ~BiTree() {
        Release(root);
    }
};

```

- A、上列 C++ 代码适用于构造各种二叉树      B、代码 struct BiNode 用于构造二叉树的节点  
 C、代码 BiTree(){root=Creat();} 用于构造二叉树      D、析构函数不可以省略

正确答案：D

5、基于第 4 题的定义，有关下面 C++ 代码的说法正确的是？（ ）

```

void Order(BiNode *bt) {
    if (bt == nullptr)
        return;
    else {
        cout << bt->data;
        Order(bt->lchid);
        Order(bt->rchid);
    }
}

```

- A、代码中 Order() 函数是中序遍历二叉树的方法  
 B、代码中 Order() 先访问根节点，然后对左子树进行前序遍历，再对右子树前序遍历  
 C、代码中 Order() 先访问中序遍历左子树，然后访问根节点，最后则是中序遍历右子树  
 D、代码中 Order() 先后序遍历左子树，然后后序遍历右子树，最后访问根节点

正确答案：B

6、有关下面 C++ 代码的说法正确的是？（ ）

```

typedef struct LinkList {
    int data;
    LinkList *next;
    LinkList *prev;
} LinkList, LinkNode;

bool ListInit(LinkList* &L) {
    L = new LinkNode;
    if (!L) return false;
    L->next = NULL;
    L->prev = NULL;
    L->data = -1;
    return true;
}

```

- A、上述代码构成单向链表      B、上述代码构成双向链表      C、上述代码构成循环链表      D、上述代码构成指针链表

正确答案：B

7、对 hello world 使用霍夫曼编码 ( Huffman Coding )，最少 bit ( 比特 ) 为？（ ）

A、4    B、32    C、64    D、88

正确答案：B

8、下面的 fiboA() 和 fiboB() 两个函数分别实现斐波那契数列，该数列第 1、第 2 项值为 1，其余各项分别为前两项之和。下面有关说法错误的是？（ ）

```
int fiboA(int n) {
    if(n == 0)
        return 1;
    if(n == 1)
        return 1;
    else {
        return fiboA(n - 1) + fiboA(n - 2);
    }
}

int fiboB(int n) {
    if((n == 0) || (n == 1)) {
        Fibob[n] = n;
        return n;
    } else {
        if(Fibob[n] == 0) {
            Fibob[n] = fiboB(n - 1) + fiboB(n - 2);
        }
        return Fibob[n];
    }
}
```

- A、fiboA() 采用递归方式实现斐波那契数列    B、fiboB() 采用动态规划算法实现斐波那契数列  
C、当 N 值较大时，fiboA() 存在大量重复计算    D、由于 fiboA() 代码较短，其执行效率较高

正确答案：D

9、有关下面 C++ 代码不正确的说法是？（ ）

```
int Depth(BiTree T) {
    if (T == NULL) {
        return 0;
    } else {
        int m = Depth(T->lchild);
        int n = Depth(T->rchild);
        if (m > n) {
            return m + 1;
        } else {
            return n + 1;
        }
    }
}
```

- A、该代码可用于求解二叉树的深度    B、代码中函数 Depth() 的参数 T 表示根节点，非根节点不可以作为参数  
C、代码中函数 Depth() 采用了递归方法  
D、代码中函数 Depth() 可用于求解各种形式的二叉树深度，要求该二叉树节点至少有 left 和 right 属性

正确答案：B

10、下面有关树的存储，错误的是？（ ）

- A、完全二叉树可以用 list 存储    B、一般二叉树都可以用 list 存储，空子树位置可以用 None 表示

C、满二叉树可以用 list 存储 D、树数据结构，都可以用 list 存储

正确答案：D

11、构造二叉树 [1, 2, 3, null, 4] ? ( )

A、1(2(4))(3) B、1(2(3)(4)) C、(1,2(3),(4)) D、(1,(2)(3),(4))

正确答案：A

12、下面有关布尔类型的函数的说法，正确的是？( )

A、bool 类型函数只能返回 0 或者 1 两种值 B、bool 类型函数可以返回任何整数值 C、bool 类型函数必须有参数传递 D、bool 类型函数没有返回值

正确答案：A

13、通讯卫星在通信网络系统中主要起到( )的作用？

A、信息过滤 B、信号中继 C、避免攻击 D、数据加密

正确答案：B

14、小杨想编写一个判断任意输入的整数 N 是否为素数的程序，下面哪个方法不合适？( )

A、埃氏筛法 B、线性筛法 C、二分答案 D、枚举法

正确答案：C

15、内排序有不同的类别，下面哪种排序算法和冒泡排序是同一类？( )

A、希尔排序 B、快速排序 C、堆排序 D、插入排序

正确答案：B

【判断题】(每题 2 分)

16、在面向对象中，方法在 C++ 的 class 中表现为 class 内定义的函数。( )

正确答案：正确

17、C++ 类的定义中，可以没有构造函数，会给出默认的构造函数。( )

正确答案：正确

18、如果某个 C++ 对象 (object) 支持下标运算符 (方括号运算符)，则该对象在所对应 class 中以成员函数的形式进行了重载。( )

正确答案：正确

19、深度优先搜索 (DFS, Depth First Search 的简写) 属于图算法，其过程是对每一个可能的分支路径深入到不能再深入为止，而且每个节点只能访问一次。( )

正确答案：正确

20、哈夫曼编码 (Huffman Coding) 具有唯一性，因此有确定的压缩率。( )

正确答案：错误

21、在下面 C++ 代码中，由于删除了变量 ptr，因此 ptr 所对应的数据也随之删除，故第 8 行代码被执行时，将报错。( )

```
int *ptr = new int(10); //第 5 行
```

```
cout << *ptr << endl; //第 6 行
```

```
delete ptr; //第 7 行
```

```
cout << *ptr << endl; //第 8 行
```

正确答案：错误

22、二叉搜索树查找的平均时间复杂度为  $O(\log N)$ 。( )

正确答案：正确

23、二叉搜索树可以是空树 (没有任何节点) 或者单节点树 (只有一个节点)，或者多节点。如果是多节点，则左节点的值小于父节点的值，右节点的值大于父节点的值，由此推理，右节点树的值都大于根节点的值，左节点树的值都小于根节点的值。( )

正确答案：正确

24、小杨想写一个程序来算出正整数 N 有多少个因数，经过思考他写出了一个重复没有超过  $N/2$  次的循环就能够算出来了。( )

正确答案：正确

25、同样的整数序列分别保存在单链表和双向链中，这两种链表上的简单冒泡排序的复杂度相同。( )

正确答案：正确

【编程题】

26、(本题 25 分) 闯关游戏

时间限制：1.0 s

内存限制：128 MB

#### 【问题描述】

你来到了一个闯关游戏。这个游戏总共有  $N$  关，每关都有  $M$  个通道，你需要选择一个通道并通往后续关卡。

其中，第  $i$  个通道可以让你前进  $a_i$  关，也就是说，如果你现在在第  $x$  关，那么选择第  $i$  个通道后，你将直接来到第  $x + a_i$  关（特别地，如果  $x + a_i \geq N$ ，那么你就通关了）。此外，当你顺利离开第  $s$  关时，你还将获得  $b_s$  分。

游戏开始时，你在第 0 关。请问，你通关时最多能获得多少总分？

#### 【输入描述】

第一行两个整数  $N$ 、 $M$ ，分别表示关卡数量和每关的通道数量。

接下来一行  $M$  个用单个空格隔开的整数  $a_0$ 、 $a_1$ 、……、 $a_{(M-1)}$ 。保证  $1 \leq a_i \leq N$ 。

接下来一行  $N$  个用单个空格隔开的整数  $b_0$ 、 $b_1$ 、……、 $b_{(N-1)}$ 。保证  $|b_i| \leq 10^5$ 。

#### 【输出描述】

一行一个整数，表示你通关时最多能够获得的分数。

#### 【特别提醒】

在常规程序中，输入、输出时提供提示是好习惯。但在本场考试中，由于系统限定，请不要在输入、输出中附带任何提示信息。

#### 【样例输入 1】

```
6 2
2 3
1 0 30 100 30 30
```

#### 【样例输出 1】

```
131
```

#### 【样例解释 1】

你可以在第 0 关选择第 1 个通道，获得 1 分并来到第 3 关；随后再选择第 0 个通道，获得 100 分并来到第 5 关；最后任选一个通道，都可以获得 30 分并通关。如此，总得分为  $1 + 100 + 30 = 131$ 。

#### 【样例输入 2】

```
6 2
2 3
1 0 30 100 30 -1
```

#### 【样例输出 2】

```
101
```

#### 【样例解释 2】

请注意，一些关卡的得分可能是负数。

#### 【数据规模】

对于 20% 的测试点，保证  $M = 1$ 。

对于 40% 的测试点，保证  $N \leq 20$ ；保证  $M \leq 2$ 。

对于所有测试点，保证  $N \leq 10000$ ；保证  $M \leq 100$ 。

#### 【参考程序】

```
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <algorithm>
#include <string>
#include <map>
#include <iostream>
#include <cmath>
using namespace std;
const int N = 10005;
const int M = 105;
```

```

const int inf = 0x3f3f3f3f;
int a[M], b[N], f[N];
int main() {
    int n, m;
    scanf("%d%d", &n, &m);
    for (int i = 1; i <= m; i++)
        scanf("%d", &a[i]);
    for (int i = 0; i < n; i++)
        scanf("%d", &b[i]);
    memset(f, -0x3f, sizeof(f));
    f[0] = 0;
    for (int i = 1; i < n; i++)
        for (int j = 1; j <= m; j++)
            if (i - a[j] >= 0)
                f[i] = max(f[i], f[i - a[j]] + b[j]);
    int ans = -inf;
    for (int i = 0; i < n; i++)
        for (int j = 1; j <= m; j++)
            if (i + a[j] >= n) {
                ans = max(ans, f[i] + b[j]);
                break;
            }
    cout << ans << endl;
    return 0;
}

```

## 27、( 本题 25 分 ) 工作沟通

时间限制：1.0 s

内存限制：128 MB

### 【问题描述】

某公司有  $N$  名员工，编号从 0 至  $N - 1$ 。其中，除了 0 号员工是老板，其余每名员工都有一个直接领导。我们假设编号为  $i$  的员工的直接领导是  $f_i$ 。

该公司有严格的管理制度，每位员工只能受到本人或本人直接领导或间接领导的管理。具体来说，规定员工  $x$  可以管理员工  $y$ ，当且仅当  $x = y$ ，或  $x = f_y$ ，或  $x$  可以管理  $f_y$ 。特别地，0 号员工老板只能自我管理，无法由其他任何员工管理。

现在，有一些同事要开展合作，他们希望找到一位同事来主持这场合作，这位同事必须能够管理参与合作的所有同事。如果有多名满足这一条件的员工，他们希望找到编号最大的员工。你能帮帮他们吗？

### 【输入描述】

第一行一个整数  $N$ ，表示员工的数量。

第二行  $N - 1$  个用空格隔开的正整数，依次为  $f_1$ 、 $f_2$ 、.....、 $f_{(N-1)}$ 。

第三行一个整数  $Q$ ，表示共有  $Q$  场合作需要安排。

接下来  $Q$  行，每行描述一场合作：开头是一个整数  $m$  ( $2 \leq m \leq N$ )，表示参与本次合作的员工数量；接着是  $m$  个整数，依次表示参与本次合作的员工编号（保证编号合法且不重复）。

保证公司结构合法，即不存在任意一名员工，其本人是自己的直接或间接领导。

### 【输出描述】

输出  $Q$  行，每行一个整数，依次为每场合作的主持人选。

### 【特别提醒】

在常规程序中，输入、输出时提供提示是好习惯。但在本场考试中，由于系统限定，请不要在输入、输出中附带任何提示信息。

### 【样例输入 1】

5

0 0 2 2

3

2 3 4

3 2 3 4

2 1 4

【样例输出 1】

2

2

0

【样例解释 1】

对于第一场合作，员工 3、4 有共同领导 2，可以主持合作。

对于第二场合作，员工 2 本人即可以管理所有参与者。

对于第三场合作，只有 0 号老板才能管理所有员工。

【样例输入 2】

7

0 1 0 2 1 2

5

2 4 6

2 4 5

3 4 5 6

4 2 4 5 6

2 3 4

【样例输出 2】

2

1

1

1

0

【数据规模】

对于 50%的测试点，保证  $N \leq 40$ 。

对于所有测试点，保证  $3 \leq N \leq 300$ ； $Q \leq 100$ 。

【参考程序】

```
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <algorithm>
#include <string>
#include <map>
#include <iostream>
#include <cmath>
#include <vector>
using namespace std;
const int N = 305;
int fa[N], dep[N];
bool vis[N];
vector<int> ch[N];
int getdep(int x) {
    return x == 0 ? 0 : getdep(fa[x]) + 1;
```

```

}
void dfs(int x) {
    vis[x] = 1;
    for (int y : ch[x])
        dfs(y);
}
bool check(int x, int n, const vector<int> &vec) {
    for(int i = 0; i <= n; i++)
        vis[i] = 0;
    dfs(x);
    for (int y : vec)
        if(! vis[y])
            return 0;
    return 1;
}
int main() {
    int n;
    scanf("%d", &n);
    for(int i = 1; i < n; i++) {
        scanf("%d", &fa[i]);
        ch[fa[i]].push_back(i);
    }
    for(int i = 1; i < n; i++)
        dep[i] = getdep(i);
    int q;
    scanf("%d", &q);
    while(q --) {
        int m, mnd = n + 1;
        scanf("%d", &m);
        vector<int> vec(m);
        for(int i = 0; i < m; i++) {
            scanf("%d", &vec[i]);
            mnd = min(mnd, dep[vec[i]]);
        }
        for (int i = n - 1; i >= 0; i--)
            if (dep[i] <= mnd && check(i, n, vec)) {
                printf("%d\n", i);
                break ;
            }
    }
    return 0;
}

```